RASTER TO VECTOR CONVERSION OF POLYGONS FOR A COLOR IMAGE SCANNER AND INK JET PLOTTER SYSTEM

Åke E. Nygren

DEPARTMENT
OF
ELECTRICAL MEASUREMENTS

INSTITUTIONEN FÖR ELEKTRISK MÄTTEKNIK
LUNDS TEKNISKA HÖGSKOLA - LUNDS UNIVERSITET
LUND - SWEDEN

1983

RASTER TO VECTOR CONVERSION OF POLYGONS

FOR A COLOR IMAGE SCANNER

AND

INK JET PLOTTER SYSTEM

Åke E. Nygren

Report 3/1983

Department of Electrical Measurements
Lunds Institute of Technology

Lund, Sweden

LUTEDX/(TEEM-1020)/1-15/(1983)

UNIV.
LEIDEN
BIBL.

SUMMARY

Digitization of color images by a drum scanner results in a raster format representation of the image. To manipulate certain parts of such a digitized raster image with existing software packages, the image objects to be manipulated have to be available in vector format. The present paper describes a method that allows the transfer of polygons in the raster image to vector format and shows the use of this software in a problem from community planning.

CONTENTS

	Page
1 Introduction	1
2 Conversion method	2
2.1 Coding of raster images	3
2.2 The chain-link coding method	3
3 Software implementation	5
3.1 The location of the polygon	5
2.2 The transformation from raster to vector format	7
4 Application of the software	8
5 Processing time considerations	14
References	15

1 Introduction

The growth of computer output devices generating images in color and black & white is mainly due to the fact that images can present complicated relations in a compressed and easily understandable way. As an example the reader is reminded of the fact that the result from a survey of the magnetic field over an area of the earth is easier to evaluate if the measured values are overlaid in different colors on a map over that area, than if they are presented in numerical form.

When presenting geographically related information, such as in the example above, it is a major problem to introduce a map over the area into the computer memory. One method is to use a digitizing table and input all the required information manually. This is a timeconsuming and expensive method especially if many details must be present in the background map (base map). Therefore the color method described above is often not used for the presentation of survey data since it is too expensive to generate the base map.

Another approach is to draw or print the base map separately without computer assistance and fasten this map in place of the recording paper on the computer output unit, i.e. a plotter, and add the computer generated/calculated information on top of it.
This method has two drawbacks:

- It separates the data from the base map which is a problem if a new copy of the survey is to be made at a later time.

- The positioning of the base map in the plotter is carried out by hand which can result in a position error of the survey data relative to the base map.

In spite of this, the method is adequate in several applications.

To facilitate the generation of a base map residing in the computer system, and make it less costly for the users, a color image scanner was developed at the Department of Electrical Measurements (Nilsson and Nygren 1981 A & B, Nilsson 1981, Nygren 1983). The scanner reads a continuous-tone image, transforms it into a bilevel raster form and stores it as digital information on a magnetic tape. Thereafter additional information, e.g. measured data, can be added to this raster image by using appropriate software in a host computer system.

The advantage of using the scanner as opposed to manual digitization is the shorter time needed to generate the base map which results in lower cost. This is especially true if the base map contains complex details.

The drawback of this scanner method is the fact that closed polygons, lines, etc are all stored in raster format and cannot be accessed or manipulated directly as discrete objects (entities). Hence special coloring of the polygon, e.g. for the generation of a thematic map, is practically impossible to achieve.

To minimize this problem, and make it possible to handle a closed polygon as an entity, a semi automatic method to find and trace borderlines of closed polygons was written. The output from this program represents the polygon in

vector form and is stored on a data file. This file can be used by other programs to color the polygon surface or whatever is wanted in the particular application. Storing the information on a file in vector form allows its manipulation with raster software packages such as COLOR, UNIRAS and MULTIRAS (Jern 1978, European Software Contractors A/S 1982)

In the following chapters the ideas behind the method are first given and then the adoption of this method in our system is described. The end of this report includes a step by step description of an application using this program.

2 Conversion method

A method to transform a closed raster polygon into a list of vectors describing the border line can be divided in three steps:

- find polygon in image area and define a point within the polygon area
- find edge of polygon and trace it
- transform raster edge information into vector representation so that the polygon can be handled as an entity

The first step can be achieved interactively or non-interactively. The interactive method uses a graphic work station with a CRT to display the image and a joystick, mouse or tablet to mark the location of the polygon within the image area. This method is easy to use, but the cost of a graphic work station is fairly high.

In the non-interactive method, the polygon location is manually calculated and entered into the computer as numerical values by an ordinary alfanumerical terminal. This method is inexpensive, as no special hardware is needed.

Another non-interactive method would be to find all polygons within the image automatically. This is not suitable in our application as the number of polygons to be transformed will often be far less than the number of polygons present in the image.

The second step, to find the edge, is a simple pixel by pixel movement in a predefined direction which is started from the point defining the polygon. This movement continues until the edge is reached. When the edge of the polygon is reached, the address of this pixel is known. Thereafter the position of all borderline pixels can be found by literally using the same method as children use when they are walking down a street with their parents - one foot on the sidewalk and one foot on the street. If the polygon is large, several thousand pixels are included in the borderline.

Finally, the third step, to transform the raster edge information into a format which makes it possible to handle the polygon as an entity, is the main issue of this paper. The polygon can be described by a table with addresses to all pixels located on the polygon borderline. This method requires a large amount of data to describe each polygon and hence the time and cost for the handling of the polygons will be high. Therefore a method to code the pixel information to reduce the amount of data is required. Thus the present

problem is closely related to image compression methods and therefore some of these methods will be reviewed in the following.

2.1 Coding of raster images

The goal of coding raster images is to reduce the number of bits required to represent the image. Several methods exist which employ different approaches to achieve a good compression ratio between the original raster image and the compressed image (Jain 1981). Two different approaches to achieve this compression are commonly used, namely:

- the runlength coding method.
  Every "run" of equal elements in the image is represented by an element code and the number of such consecutive elements. An element can be a single pixel or a pattern of several pixels.

  The runlength coding method gives the best results in images with few details, as this produces long runs of equal elements. In runlength coding and similar coding methods based on repetitive patterns no information on single objects in the image is extracted.

- the coding of objects method.
  The image reduction is achieved by coding the objects within the image. The RC-chain code method (Cederberg 1979) and similar methods extract line information on a scan line by scan line basis and link the line segments together to form the polygons or lines that are present in the image. The method used to describe each of the line segments is based on the chain-link coding method or, as it is often referred to, the Freeman coding method (Freeman 1961).

In our application, as stated in the beginning of this chapter, only selected polygons are to be transformed. Hence only the method to code the individual line segments, used in the coding of objects method described above, is needed for the present purpose.

2.2 The chain-link coding method

The chain-link coding method is suitable when the borderline of a closed polygon is to be traced. The curve is followed clockwise or counter clockwise until the first encountered pixel in the border line is reached again. Only one code for each move is saved.

In the method a move is made from any given point to the next point on the boundary, the move being selected among a set of predefined directions and distances. The number of directions and distances can vary considerably as shown in Fig. 2-1.

Thus a curve in an image can be described by a (X0,Y0) starting coordinate and a sequence of numbers, where each number represents the "direction and distance" code. In a raster image each pixel has 8 adjacent neighbours. Therefore to avoid quantization errors in the tracing of the curve, it is necessary to use at least 8 directions (Groen and Verbeek 1978). Hence the 8-point code given in Fig. 2-1 was chosen.

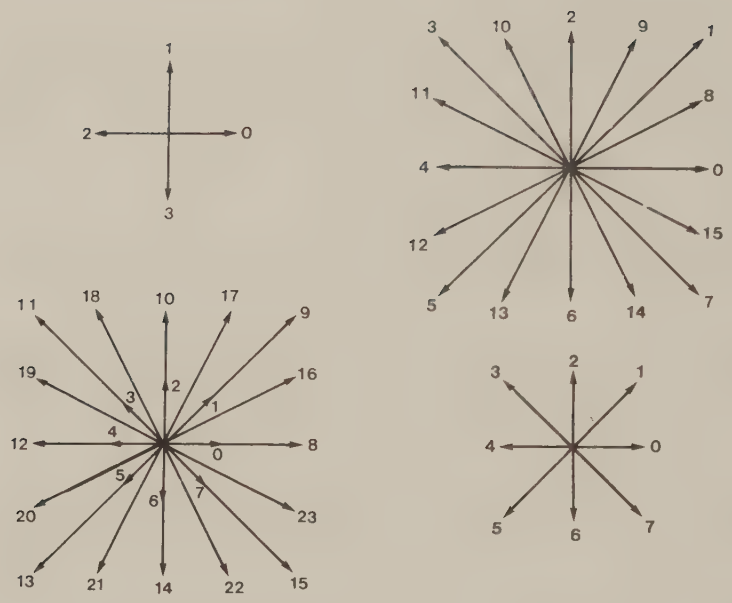

Fig. 2-1 Possible 4-, 8-, 16-,and 24-point chain codes. The 8-point chain code is used in this application, as the number of adjacent pixels to a pixel in a raster image is 8.

3 Software implementation

The implementation of the software system at the Department of Electrical Measurements is based on the non-interactive method as there was no suitable graphic station available for the interactive method.

To facilitate the manual measurements of the polygon location within the raster image, a program was written that places a 10 mm grid net over the image for measuring purposes. The raster image with the grid net superimposed was thereafter plotted on the color ink jet plotter (Hertz and Månsson 1972, Smeds 1973, Bladh 1982). The size of the raster image was set according to the scanner image format 670*614.4 mm (3350*3072 pixels) as the color scanner is assumed to be the normal raster image input unit.

The implementation of the chain-link coding method together with methods for finding the polygon and it's edge was straightforward. In the following the main parts of the program will be described in a step by step fashion, the same way the program executes the transformation as shown in Fig. 3-1. The selection of input/output files etc, which is standard Fortran 77 procedure will not be discussed.

3.1 The location of the polygon

The X and Y coordinate are entered from the alfanumerical terminal. The values are given in mm (XMM,YMM), and internally transformed to number of pixels (XPIX,YPIX). The program also asks for an identification number to be associated with the polygon (POLYID)

The raster image to be operated on is stored on a disc file as several small rectangles (XWIDE*YHIGH pixels). The main memory of the host computer can only hold one subset of all the rectangles at a time, so the program must calculate the disc address to the appropriate rectangles and retrieve them before any processing of the image can take place. In the present version a strip containing 50*3072 pixels is simultaneously present in the main memory. This is the equivalent of 50 scan lines.

As each pixel is represented by a bit in the computer, the address of any pixel differs in the X and Y directions. Each data byte in the computer contains 8 consecutive pixels in the Y direction, while 8 consecutive pixels in the X direction are stored in 8 different bytes which are separated by 384 bytes each. The number 384 is the length of each scan line in the Y direction which is 3072 bits (384 bytes) long.

As shown in Fig. 3-1, the program tests if the pixel at the (XPIX,YPIX) location is part of a line or not. If this is not the case, YPIX is decremented by 1 and the test is repeated.

When a polygon edge is reached, the XPIX and YPIX values are stored in a virtual array VECT that will contain all vectors of the present polygon. The present direction, decreasing in Y direction, is called 6 and it is temporarily stored for comparison with the direction to the next pixel on the polygon borderline. The codes for different directions used in this report are given in Fig. 2-1.

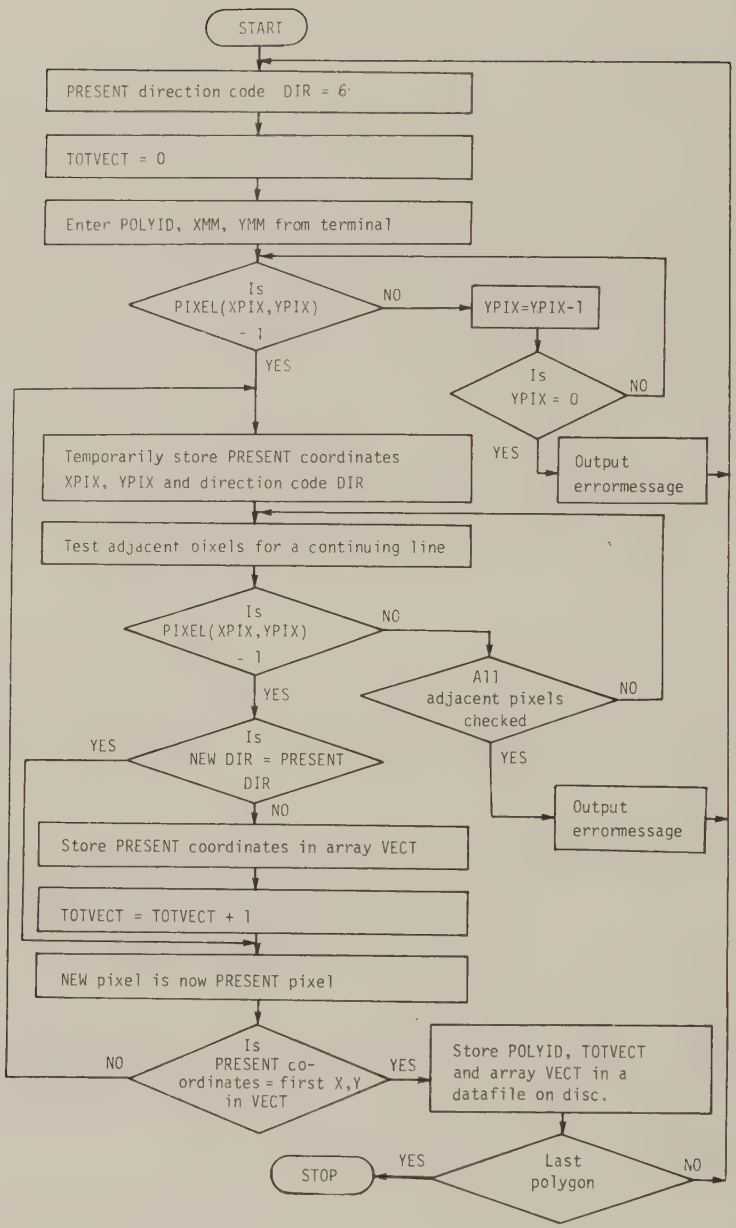

Fig. 3-1 Flow chart describing the raster to vector conversion program.

If no polygon edge is found due to error in coordinates input from the terminal, the program writes an error message on the terminal when YPIX reaches the edge of the image, and asks for a new pair of coordinates.

3.2 The transformation from raster format to vector format

The chain-link coding method, as described in chapter 2.2, stores the (X0,Y0) coordinate, and the direction codes to all the following coordinates in the curve. To read and understand the information that results from this requires the access to the chain-link codes used. This puts a heavy constraint on the use of said information, especially when it shall be used in conjunction with other programs. Therefore, this metod was modified to store coordinates instead of codes, and to further compress the information only the first point of every straight line in the directions 0-7 was stored. As all the straight lines are linked together in a closed polygon, the first point of one straight line is at the same time the last point of another straight line.

In the following an example of how the transformation is achieved will be given.

The XMM and YMM values entered from the terminal are converted from mm into pixel units (XPIX,YPIX), and the part of the image containing the (XPIX,YPIX) address is loaded from the disc. The black borderline is found by testing the pixels while decreasing the YPIX value. When a hit occurs, we know that the pixel must be located on the inside of a border line of unknown width and length encircling the polygon. The X and Y coordinates of the present pixel, as well as the direction code (0 through 7) used to reach it are temporarily stored. According to the direction codes given in Fig. 2-1, the first pixel must have the direction, DIR = 6. This pixel is now called PRESENT. Next black pixel is tested for among the 7 neighbours not previously visited. That is all surrounding pixels except the pixel in DIR = 2.

As the present direction is 6, and the movement along the inner side of the borderline is counterclockwise, the first neighbour to be examined is the one found in direction DIR = 1 and thereafter in a clockwise order 0, 7, 6, 5, 4 and 3 until the next black pixel is found. The direction to and the coordinates of the NEW pixel are temporarily stored. The direction to the NEW pixel is compared with the direction to the PRESENT pixel. If they are equal, the X and Y value of the PRESENT pixel are discarded. If they are different, the PRESENT pixel coordinates are saved in array VECT and the variable TOTVECT is incremented by 1. TOTVECT contains the number of coordinate pairs stored in VECT.

The NEW pixel is now the PRESENT pixel and the whole search is repeated. Before each test for a NEW pixel, the program checks that the pixel is present in the computer memory. When the edge of the 50 scan line strip is reached, the program automatically retrieves the following 50 scan lines from the disc.

When the whole polygon is encircled, the polygon identification number, POLYID and the number of coordinate pairs, TOTVECT, are written on a disc file immediately followed by all coordinate pairs VECT.

The program asks for the next polygon to be transformed, and the procedure above is repeated. When the last polygon has been encircled, the program terminates and all transformed polygons are sequentially stored on the same disc file in the format described above.

4 Application of the software

This is a short step by step description on how to use the raster to vector converter program. The example uses a simulated black and white map over a community and the intention is to be able to color the different building blocks according to data fed into the computer.

The map is converted and stored on a magnetic tape through the use of the color image scanner (Nygren 1983). This tape can be used directly by the UNIRAS software system to generate a base map. The original map drawn by hand is shown in Fig. 4-1.

A 10 mm grid net is added to the raster image through the utility program SCANMEASURE and a hard copy of the map with the grid net is plotted on the color ink jet plotter for measurement purposes. The map with the grid net superimposed is shown in Fig. 4-2.

The raster to vector conversion program, RASVECT, is started and asks for the name of the file containing the raster based map generated by the color image scanner and the name of the resulting vector file. Thereafter the conversion is handled as described in chapter 3, and all polygons (building blocks) are stored in the vector file. Each polygon has been given an identification number, POLYID, which makes it possible to retrieve the correct polygon when needed. The results from the RASVECT program stored in a vector file are given in Fig. 4-3. Only the border line of each polygon is shown, and the polygon identification, POLYID, is written in each of the polygons. The following table gives the number of points necessary to represent each one of the polygons.

Polygon identity	Number of points
1	124
2	53
3	85
4	58
5	76
6	62
7	78
8	40
9	128
10	205
11	252
12	130
13	158
14	162
15	89

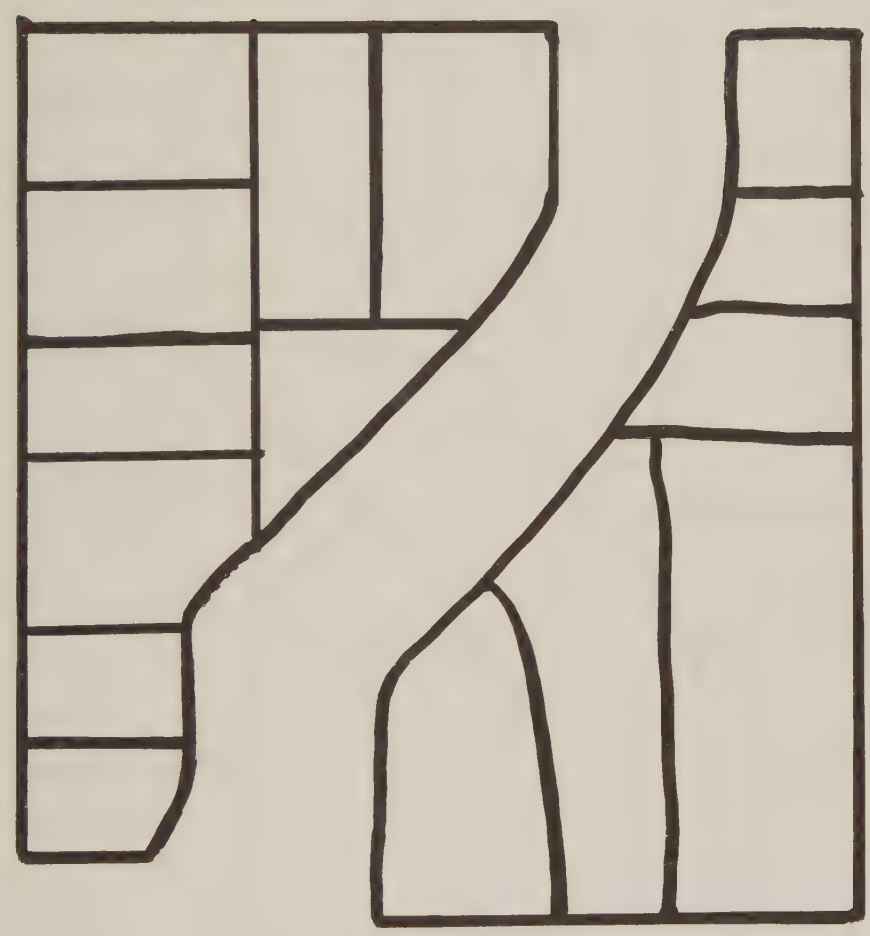

Fig. 4-1 Simulated map of a community area depicting a few building blocks. This original was scanned by the color image scanner.

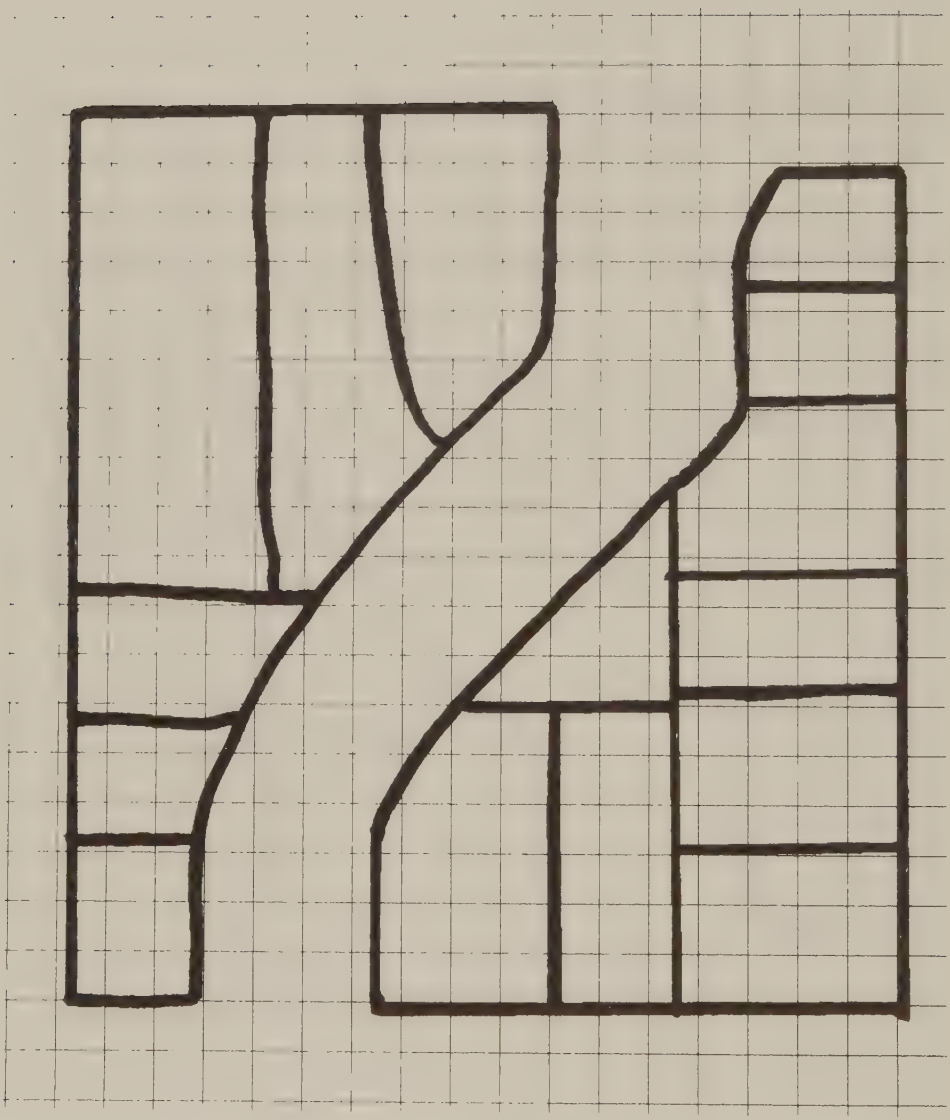

Fig. 4-2 Simulated map of a community area. A 10 mm grid net is superimposed on top of the map as aid in estimating a coordinate pair in each of the building blocks.

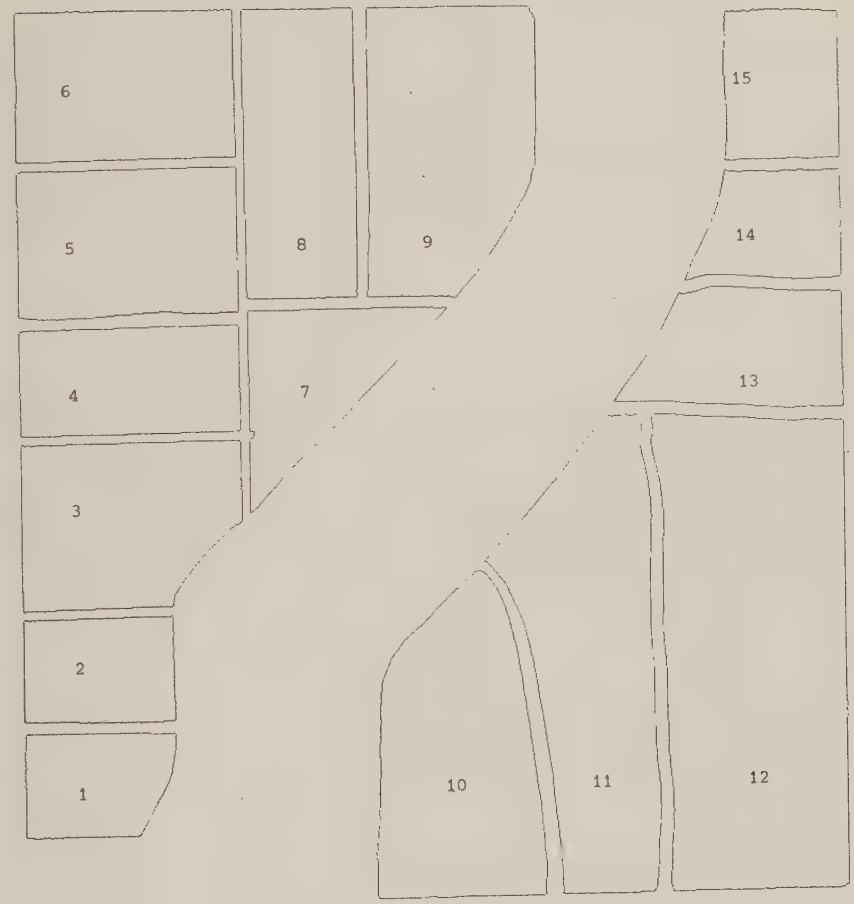

Fig. 4-3 Simulated map of a community area. The building blocks have been outlined by the technique described in this paper.

To add color or gray-tone information to the raster image, the UNIRAS software system has been used. The program which calculates the color information to put on a particular polygon reads the data file containing polygon borderline information in vector format.

All reads to the vector file are standard Fortran read statements. An example of how these reads can be handled is given below:

```
      DIMENSION XVECT(6500),YVECT(6500)
      INTEGER POLYID,TOTVECT
      *
      OPEN(UNIT=48, FILE='filename',STATUS='old',ERR=...)
      *
      READ(48,1)POLYID,TOTVECT
      DO 10 I=1,TOTVECT
      READ(48,2)XVECT(I),YVECT(I)
10    CONTINUE
      *
1     FORMAT(I4)
2     FORMAT(2F6.2)
      *
      END
```

Before any information is read from the vector file, the program calculates the colors that each polygon will be given. Thereafter the program reads the first value in the data file (POLYID) to determine which one of the polygons it has encountered. It thereafter reads the second value from the data file (TOTVECT) to determine how many coordinate pairs (VECT) that will follow. The program stores the following TOTVECT coordinate pairs, and makes a subroutine call to the GSURF routine in the UNIRAS software.
The call has the following format:

```
CALL GSURF(XVECT,YVECT,TOTVECT,ICOLOR,IFRAME)
```

XVECT and YVECT are arrays which contain the coordinate information from the data file
TOTVECT = number of points
ICOLOR = Color index of the shade used to fill the polygon surface. The possible color shades are given in the RASPAK User manual (European Software Contractor 1983)
IFRAME = Width of frame plotted around the polygon (0 - 10 pixels)

The number of vectors included in the call is not limited by the software.

This is repeated until the last POLYID is found or the end of the vector file is reached.

Fig. 4-4 shows the resulting raster image. Here the building blocks have been given different shades of gray using the above mentioned program.

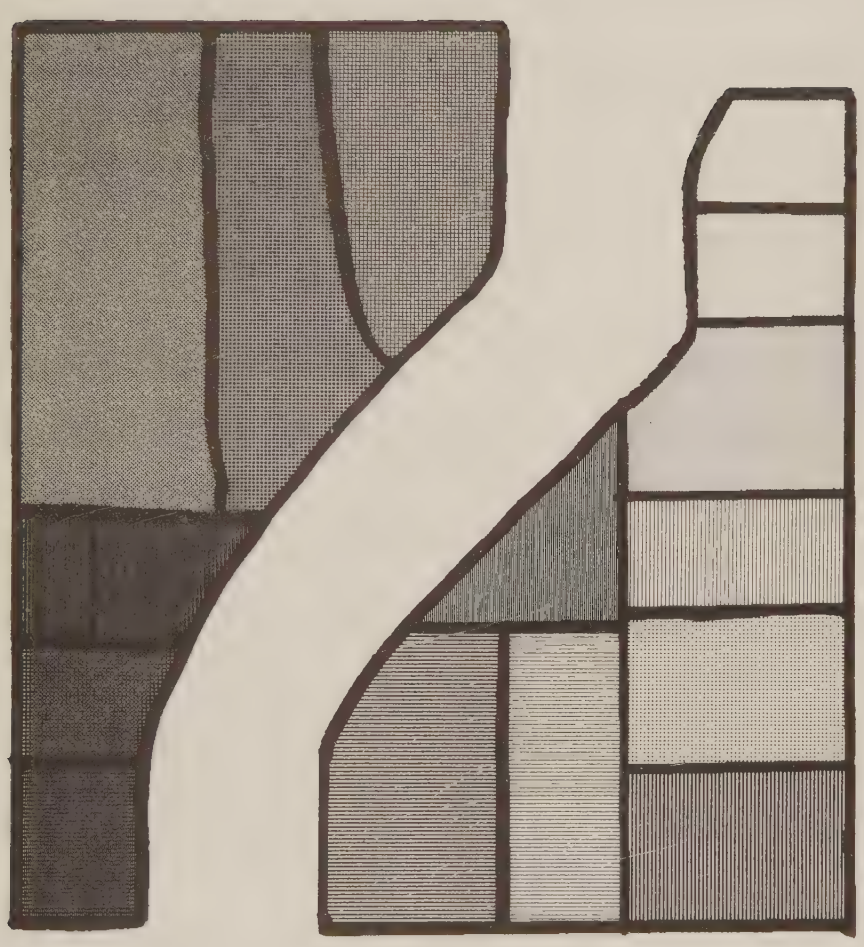

Fig. 4-4 Simulated map of a community area. The building blocks have been shaded by the technique described in this paper.

5 Processing time considerations

The processing time for the raster to vector conversion program is strongly dependent on the size and orientation of the polygon. Movements along the Y axis are fairly fast, while the movements along the X axis are slower as the program must make a disc access every time 50 new scan lines (10 mm) have been examined.

The example given in chapter 4 takes approximately 2 CPU minutes to calculate. This is very fast compared to the manual digitizing methods.

REFERENCES

Bladh, K. (1982) "The Three-Color Ink Jet Plotter - A New Device for the Presentation of Geophysical Data", Technical Report LUTEDX/(TEEM-1013), Department of Electrical Measurements, Lund Institute of Technology, pp. 1-160.

Cederberg, R.L.T. (1979) "Chain-link Coding and Segmentation for Raster Scan Devices", Computer Graphics and Image Processing 10, pp. 224-234.

European Software Contractors A/S (1982) "Raspak user's manual", Gentofte, Denmark.

Freeman, H (1961) "On the Encoding of Arbitrary Geometric Configurations", IRE Trans., EC-10, June, pp.260-268.

Groen, F.C.A and Verbeek, P.W. (1978) "Freeman-Code Probabilities of Object Boundary Quantized Contours", Computer Graphics and Image Processing 7, pp. 391-402.

Hertz, C.H. and Månsson, A. (1972) "Electrical Control of Fluid Jets and Its Application to Recording Devices", The Review of Scientific Instruments, Vol. 43, No. 3, March, pp. 413-416.

Jain, A.K. (1981) "Image Data Compression: A Review", Proceedings of the IEEE, Vol. 69, No. 3, March, pp. 413-416.

Jern, M. (1978) "Color Plotting System. Basic Software Package Version 6, Programmers Guide", Lund University Computing Center.

Nilsson, J.O. (1981) "A Color Separating Optical Scanner Head". Technical Report LUTEDX/(TEEM-1007), Department of Electrical Measurements, Lund Institute of Technology, pp. 1-37.

Nilsson, J.O. and Nygren, A.E. (1981A) "A Three Color Bilevel Scanner", Technical Report LUTEDX/(TEEM-1009), Department of Electrical Measurements, Lund Institute of Technology, pp. 1-25. Accepted for publication in Society for Information Display (SID).

Nilsson, J.O. and Nygren, A.E. (1981B) "Conversion of Continuous-tone Images into Bilevel Representation Applied to a Color Image Scanner and a Color Ink Jet Plotter", Technical Report LUTEDX/(TEEM-1008), Department of Electrical Measurements, Lund Institute of Technology, pp. 1-126.

Nygren, A.E. (1983) "A Microprocessor Controlled Color Image Scanner", Technical Report LUTEDX/(TEEM-1018), Department of Electrical Measurements, Lund Institute of Technology, pp. 1-104.

Smeds, B. (1973) "A 3-Colour Ink Jet Plotter for Computer Graphics", BIT 13, Copenhagen, pp. 181-195.